<div align="center">**Amendments to the Claims**</div>

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) In an information processing system comprising a plurality of stacks each comprising at least one stack frame, a method for associating a phase with an activation in a computer program running at least one thread, the method comprising steps of:

a) allocating space in memory for an activation count for each frame;

b) zeroing the activation count ~~wherever~~ whenever the program creates a new stack frame;

determining whether an interval has transpired during program execution;

continuing the program until the interval transpires if the interval has not transpired;

examining each stack's content and incrementing the activation count for each frame once the interval has transpired;

c) analyzing the stack for each thread and incrementing the activation count for each frame; and

d) associating ~~a~~ the phase with the activation whose activation count is non-zero.

2. (Currently amended) The method of claim 1 wherein the step of analyzing the stack ~~c)~~ further comprises logging activation counts during each interval.

3. (Original) The method of claim 1 wherein the activation count is implemented by reserving storage in each stack frame.

4. (Currently amended) The method of claim 1, further comprising performing step performing step ~~c)~~ at periodic intervals of time according to a system clock.

5. (Currently amended) The method of claim 1 further comprising examining each stack to determine the number of frames used in the stack and incrementing the activation count <u>for each frame in the stack</u> ~~by the number of frames~~.

6. (Original) The method of claim 1 further comprising ensuring that when a phase ends, an action is performed immediately.

7. (Original) The method of claim 6 further comprising changing the return address to force the program to call a designated procedure when the frame returns.

8. (Original) The method of claim 1 further comprising scheduling garbage collection after each associated phase.

9. (Currently amended) The method of claim 1 further comprising scheduling thread switches <u>after the step of associating the phase</u> at ~~phase boundaries~~.

10. (Currently amended) The method of claim 1 further comprising scheduling checkpoint operations after ~~each associated~~ <u>the step of associating the</u> phase.

11. (Currently amended) The method of claim 1 further comprising presenting a visualization of program phase behavior <u>after the step of associating the phase</u>.

12. (Currently amended) The method of claim 1 further comprising resetting profile data <u>after the step of associating the phase</u> ~~at phase transitions~~.

.

13. (Original) The method of claim 1 wherein the activation count is represented by a single bit, representing the presence or absence of a running phase.

14. (Original) The method of claim 1 further comprising implementing activation counts in a side data structure.

15. (Original) The method of claim 1 wherein the activation count is implemented as an array paralleling the stack.

16. (Currently amended) A system for associating a phase with an activation of a computer program, wherein the program supports garbage collection, the system comprising:

a plurality of stacks each comprising at least one stack frame comprising an activation counter; and a processor comprising logic for:

zeroing the activation count ~~wherever~~ whenever the program creates a new stack frame and after garbage collection is performed;

determining whether an interval has transpired during program execution;

continuing the program until the interval transpires if the interval has not transpired;

examining each stack's content and incrementing the activation count for each frame once the interval has transpired;

analyzing the stack for each thread and incrementing the activation count for each frame; and

associating a phase with an activation whose activation count is non-zero.

17. (Original) The system of claim 16 wherein the processor further comprises, logic for logging activation counts during each interval.

18. (Original) The system of claim 16 wherein the activation count is implemented by reserving storage in each stack frame.

19. (Original) The system of claim 16 wherein the processor further comprises logic for analyzing the stack for each thread and incrementing the activation count for each frame at

periodic intervals of time according to a system clock.

20. (Original) The system of claim 16 wherein the processor comprises logic for causing the system to call a designated procedure when the frame returns.

21. (Original) The system of claim 16 wherein the memory comprises instructions for causing the system to call a designated procedure when the frame returns.

22. (Currently amended) The system of claim 16 wherein the processor comprises logic for examining each stack to determine the number of frames used in the stack and incrementing the activation count <u>for each frame in the stack</u> ~~by the number of frames~~.

23. (Currently amended) A computer readable medium comprising program instructions for:

  ~~a)~~ allocating space in memory for an activation count for each frame;

  ~~b)~~ zeroing the activation count ~~wherever~~ <u>whenever</u> the program creates a new stack frame;

  <u>determining whether an interval has transpired during program execution;</u>

  <u>continuing the program until the interval transpires if the interval has not transpired;</u>

  <u>examining each stack's content and incrementing the activation count for each frame</u> <u>once the interval has transpired;</u>

  ~~c)~~ analyzing the stack for each thread and incrementing the activation count for each frame; and

  ~~d)~~ associating ~~a~~ the phase with the activation whose activation count is non-zero.

24. (Original) The computer readable medium of claim 23 further comprising program instructions for logging activation counts during each interval.

25. (Original) The computer readable medium of claim 23 further comprising program

6

instructions for reserving storage in each stack frame for the activation count.

26. (Currently amended) The computer readable medium of claim 23 further comprising program instructions for examining each stack to determine the number of frames used in the stack and incrementing the activation count for each frame in the stack ~~by the number of frames~~.

27. (Original) The computer readable medium of claim 23 further comprising program instructions for ensuring that when a phase ends, some action is performed immediately.

28. (Original) The computer readable medium of claim 23 further comprising program instructions for changing the return address to force the program to call a designated procedure when the frame returns.

29. (Original) The computer readable medium of claim 23 further comprising program instructions for scheduling garbage collection after each associated phase.

30. (Original) The computer readable medium of claim 23 further comprising program instructions for scheduling thread switches at phase boundaries.